COMPUTING AT SCHOOL
EDUCATE · ENGAGE · ENCOURAGE

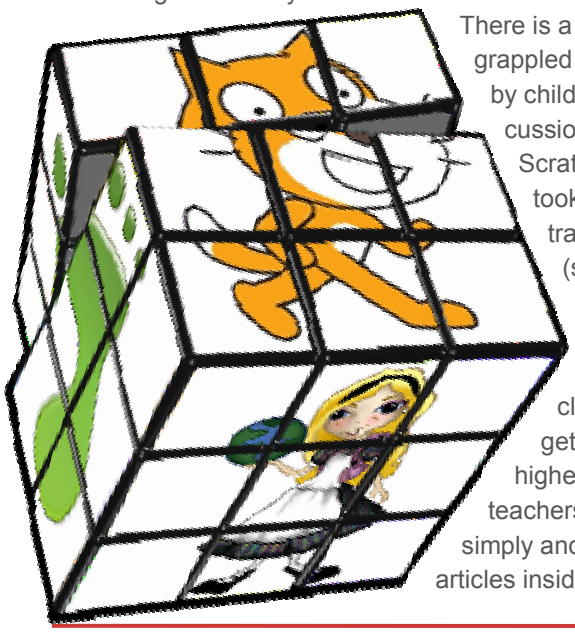
SWITCHEDON

COMPUTING AT SCHOOL NEWSLETTER SUMMER 2011

COMPUTING TEACHERS SPOILT FOR CHOICE?

CAS local hubs are growing rapidly. As word spreads about their success, more and more people are volunteering to host meetings. They provide an excellent forum to swap teaching tips and keep abreast of developments as ever increasing numbers of teachers dip their toes into teaching computing. But there's also a recurring theme that keeps being raised; what software is best to introduce young children to programming? And what's the difference between different tools? For time pressed teachers, the array of possibilities can seem daunting.

The development of visual languages such as Scratch and Alice, and child centred Integrated Development Environments such as Greenfoot have opened up all sorts of possibilities that simply weren't available even ten years ago. Of course, they are just the most well known. Add to that list powerful extensions such as Scratch BYOB, modelling tools like StarLogoTNG and game creation through GameMaker and Kodu. Throw in too, the 'traditional' educational languages such as Logo and Basic (now released as SmallBasic) and the possibilities for coding through Javascript and it's easy to see why more and more staff are asking where they should start.



There is a rich academic debate that has grappled with the barriers to coding faced by children. A good starting point is a discussion between the key developers of Scratch, Alice and Greenfoot which took place last year. You can find the transcript on Michael Kölling's blog (see the supplement for links).

What ultimately informs academic research on learning is classroom experience. Interest in getting kids coding has never been higher. CAS wants to encourage ICT teachers to give computing a go. Start simply and pick one project. We hope the articles inside inspire you to take the plunge.

HERE COMES THE SUMMER

The summer term is here, the coursework is nearly marked and soon teachers will be turning their attention to planning for next year.

The educational landscape is changing with new announcements from the government dominating the headlines. But no matter what changes finally come to the fore, one thing is certain; up and down the country more and more teachers are getting SWITCHEDON to the educational benefits of computing. CAS local hubs are spreading, discussions and debate thrive and an ever growing number of organisations are joining the call to get computing back into the curriculum.

For those in the classroom, things have never been so good. So good in fact, that the biggest problem for most staff is knowing where to start, such is the wealth of resources now at our disposal.

Inside you'll find suggestions for activities at every level, from primary to A level. You'll find details of a growing number of CPD opportunities, extra curricular activities and resources to get your pupils thinking. And in June, CAS will hold its third annual teacher conference. Places are limited so make sure you book your place soon. See back page for details.

The "Computing At School" working group (CAS) is a membership association in partnership with BCS, The Chartered Institute for IT and supported by Microsoft, Google and others. It aims to support and promote the teaching of computing in UK schools.

bcs Academy of Computing

DELIVERING A COMPUTING CURRICULUM FOR SCHOOLS

CAS exists to promote the teaching and learning of computing in schools. We believe that computing is a discipline, in the same way that maths or history are disciplines, with its own body of theory, ideas, principles, techniques, and methods. If we are going to make this claim, we need to get down to brass tacks and articulate a clear idea of what "computing" actually means. Waffle won't wash.

"Computing: a curriculum for schools" (see web supplement) is CAS's answer to that question. Modelled on the National Curriculum, its sections should look familiar:

- Importance of the subject. What territory does "computing" cover?
- Key concepts. Key themes and ideas.
- Key processes. What should computing students be able to do?
- Range and content. What should computing students know?
- Level descriptors.

It is a key document. All CAS members have had an opportunity to comment.

But it is only a start. It says *what computing is* but does not say *how it can be taught*. There is plenty of raw material available (e.g. cs4fn, CSInside, CSUnplugged), but there is still a gap between that and a cohesive scheme of work. With that in mind we have started work on a new project "Delivering a computing curriculum". Schools will have very different approaches to computing, so we need a flexible story. Our plan is to develop a set of CAS boxes, each being a 3 to 6-lesson computing topic, targeted at a specific year group. The boxes may include motivation, outline lesson plans, practical exercises, pointers to resources, assessment models, and a mapping to the curriculum document. Boxes can be combined to provide scaffolded learning opportunities and breadth of coverage of the curriculum.

Some are already doing this, initiated by the SSAT Maths & Computing head teachers. Are you willing to help? If you have a computing project that works with your pupils, please get in touch. Contact details in the supplement. Let's start by sharing what already works. *Simon Peyton-Jones*

PROMOTING COMPUTATIONAL THINKING: IT'S GOOD TO TALK

The widely acclaimed Computer Science Inside activities highlight the joy and awe of computer science, inspiring pupils to delve further. One CSInside developer reflects on his recent experiences designing a course in the USA.

How can we embed core computational thinking skills in all learners? I've been reflecting on this with my colleague at the University of California, San Diego – Dr Beth Simon. I helped Beth design and run a general education course designed to develop IT skills. Such a course could be deployed in early secondary school. Here's the stinger. It's a programming course, using Alice. Over the last decade, we've tried to avoid programming as an introduction to computing as it's seen as too hard or too specialised, but we're getting amazing responses from the students. Here's one; "Programming allows a person to think more logically – thinking in order and debugging allows the user to gain valuable problem solving skills. Aspiring to go to law school, thinking logically is extremely important and I think this has helped."

Using a program like Alice gives us valuable exposure to discussing things technically with other people and explaining clearly what we are trying to do. The core of the course is sufficient 'easy' engagement with programming concepts along with significant practice in being able to talk about those concepts. The talk is the key. If you can't talk about it, you don't understand it. In typical programming classes, we are often so focused on the final program that works or doesn't work, that we lose sight of what's really important - did the learner understand what they did? Can they describe/explain it? The product is nothing - just a vehicle for enabling the learner to practice and master the concepts. Much of the face-to-face time is spent with students discussing in triads the meat of the course. They argue and justify their viewpoints with their peers, and in so doing, we believe, strengthen their ability to "talk IT". Furthermore, the combination of extensive discussion and real programming work leads to a deep understanding of a few core computational thinking concepts:

- Computing devices do what you tell them to do (deterministic)
- They do exactly what you tell them to do (precise)
- Their operation can be analysed and understood (comprehensible)

I speculate that the typical programmer / non-programmer divide that we see so often in our classes arises because these core concepts are not in place in the so-called non-programmers, and although we may mention them, the activities of our class don't embed them viscerally in our learners. Early life experiences may embed them, but we don't. If the success of this new class can be widely repeated, then subsequent CS Inside-style sessions will have much more lasting traction with the pupils. Instead of being just outreach to engage the few, they become real teaching units because all pupils will be prepared to appreciate the algorithmic underpinning of each session. This "programming-first" model enables a truly inclusive computing education for all. Who is up for trying this format in their class?! *Quintin Cutts*

PROGRAMMING AND PEDAGOGY: LEARNING ABOUT LEARNING

The feature on Seymour Papert in last term's SWITCHEDON reminded us that the activity of learning through the construction of shared knowledge is readily experienced by students as they learn to program.

Computing gives students direct experience of other powerful approaches to learning. Papert's constructionism owes much to the constructivism of Piaget and others, which sees learners' understanding of the world as something they make for themselves through experience. For me, this is why computing must lie at the heart of IT education. Each learner comes to the computer with their own mental model of technology. Often, new experiences are readily assimilated into that model, but every once in a while (or perhaps more frequently…), the computer doesn't behave as expected. The new experience must be accommodated in a more accurate schema. The study of computing, focusing on the understanding of technology, greatly facilitates this process. The construction of computer models helps here too, as we make explicit our understanding of at least some part of the world, comparing or contrasting the virtual and the real. Douglas Adams' introduction to the ICT National Curriculum remarked, "With scientific method, we took things apart to see how they work. Now with computers we can put things back together to see how they work, by modeling complex, interrelated processes, even life itself. This is a new age of discovery, and ICT is the gateway."

The construction of meaning needn't only be the work of the lone scientist though. Vygotsky's social constructivism sees learning as a social process in which language plays a pivotal role. Learning takes place in the space between what learners can do on their own and what they can accomplish when working alongside a more knowledgeable other, inside their 'zone of proximal development'. It's perfectly valid for a classmate to take on this role. Shared access is undoubtedly a significant factor in Sugata Mitra's successes with 'hole in the wall' learning in India and Self Organised Learning Environments in Gateshead. The 'pair programming' approach to agile development provides a practical methodology for exploring at least some aspects of social constructivism in practice. The craft of programming also provides insights into the community of practice ideas of Etienne Wenger, through the legitimate peripheral participation and informal apprenticeships served in open source and game development communities, rather than as lone students. The opportunities to build on, extend and add rigour to students' informal learning beyond school, whilst not unique to computing, are a key factor in successful and stimulating teaching.

A computing curriculum also offers much that helps make sense of learning throughout and beyond school: artificial intelligence / machine learning give pupils a chance to think about thinking and to learn about learning. Systems thinking and the blend of creativity, logic and attention to detail that good programming requires have wide application that stretch far beyond the subject itself. *Miles Berry*

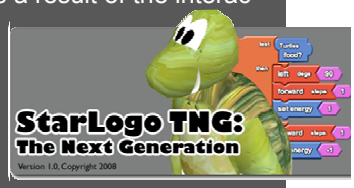
CREATIVE PROBLEM SOLVING AT KS3: CAS AT BETT

The ideas in these articles were the subject of a well attended seminar and subsequent discussion at BETT, where the concepts behind the building blocks of knowledge, were linked to examples of problem solving activities in BYOB, App Inventor for Android and StarLogoTNG.

MODELS, SIMULATIONS AND DYNAMIC SYSTEMS

For thousands of years people have been creating models to help understand the world around them. Creating models not only fosters building skills but develops a deeper understanding of the concepts embedded in the model. Think of a child and a paper airplane, or bridge building in Lego.

StarLogoTNG allows novices to build and explore their own computer models, and develop a deeper understanding of patterns and processes. Developed by the team at MIT Media Lab, who brought us Scratch, StarLogoTNG uses the same visual blocks approach. It is designed specifically to allow the study of decentralised systems, where patterns arise from the interaction of lots of individual agents. Decentralised systems are very common. Think of flocks of birds, shoals of fish or colonies of ants. Group behaviour emerges as a result of the interactions of individual behaviour.



StarLogoTNG allows pupils to explore such emergent behaviour. Simple rules can be created and applied to hundreds or even thousands of agents. The emerging patterns, or behaviour have an intrinsic 'wow' factor for pupils. There are endless project possibilities well within the capabilities of Key Stage 3 pupils. One prerequisite however is a good grasp of the concept of procedures and variables. As such, prior experience using, say, BYOB allows pupils to focus more on what they are trying to model, rather than the mechanics required to do it. The website provides plenty of ideas to whet the appetite. Further details and supporting material to model the spread of an epidemic in the supplement. *Roger Davies*

MATHEMATICAL PRINCIPLES BEHIND COMPUTER SECURITY

What is computer science? It's a question often asked by sixth form students considering their degree choices, many of whom may never have encountered anything other than ICT at school. Daniel Page and Nigel Smart, from Bristol University, have set out to illuminate the subject in a wonderful book, freely available from their website. Taking the theme of 'Information Security' they guide the reader through familiar terrain, such as virus checking and encryption, using each short chapter to introduce the computer science that lies behind the topics. More specifically, they focus on introducing some of the mathematical concepts involved.

Philip Jackson is typical of many students, having studied Computing, but not Maths at A level. In preparation for his Computer Science degree, he has been working through the book this year. "I found it very interesting", reports Philip. "It gave a more formal, mathematical definition to things which in secondary school had always been explained in words, particularly stuff like argorithmics, boolean operations, matrices and modular arithmetic. The language is clear and diagrams and examples are used very well throughout. It has a logical flow and doesn't jump into things that require background knowledge without first explaining it. There are several topics relevant to A level such as Hamming Code. There are practical activities too and the BASH appendix is the best starting point for a noob I've found."

This book is a gem, not just for computing students, but also those studying maths A level who may be developing an interest in computing. If, like me, you are a computing teacher and don't have a mathematical background, this book is a must. It will enrich your understanding of many relevant areas. The authors would appreciate your feedback. Their contact details are in the supplement. *Roger Davies*

USING SCRATCH AND ROBOMIND TO NAVIGATE HAMPTON COURT

The Digital Schoolhouse and Hampton Court Palace have collaborated to develop two computing lessons aimed at KS2 and 3 children to offer something new to the Hampton Court Palace educational visitors.

Langley Grammar School's ICT & Computing Department has adapt some existing Digital Schoolhouse learning activities that will be available free from the Hampton Court Palace website when the programme launches in June. To ensure lessons are fully accessible, kinaesthetic learning activities (Human Robots and a magic trick) are used to first teach the ideas of sequencing, accuracy, repeat and selection statements. KS2 pupils use a carpet with the famous Hampton Court maze on it to act out instructions. KS3 pupils map the decision points in the maze using the carpet as a guide before testing their work in the maze itself. They are given a wooden block with paint representing the robot sensors. Pupils use these to complete two challenges. The first gets the robot making simple left and right turns. The second helps pupils understand they need to create a nested IF statement with left and right turns. Pupils thus develop skills to instruct their robot to navigate an unfamiliar environment independently. They write the instructions in shapes matching commands in Scratch. Using Scratch shapes makes it easier to experiment. Pupils can move the shapes provided back and forth to test their ideas. The Palace uses Scratch to test the answers on an example maze, similar to lessons offered at the Digital Schoolhouse.

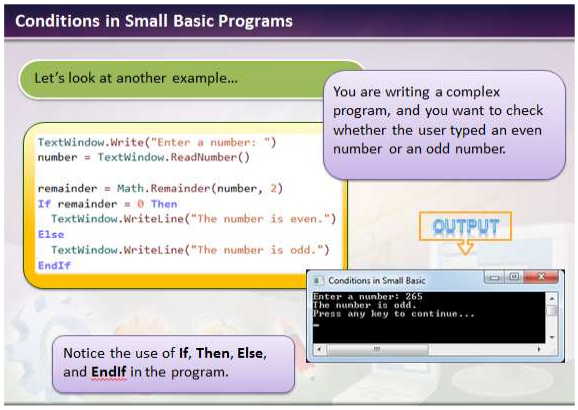
So how does it work? Using sensors the robot is aware of its environment. If you ask, it can check to see where objects might be. To ensure the 'wall following' robot stays on path we ask it to check for a wall on the right side and follow that wall. If there is a wall to the right but forwards is NOT clear it turns left. When the robot encounters a right hand bend the wall disappears, so a second conditional clause is needed to instruct the robot to turn right and move forward one square. The full wall follower procedure is described in the supplement. The activities and Scratch make it easier for pupils to create code in Robomind to mirror the instructions as all the pupils have to worry about is getting the correct syntax. Pupils have the opportunity to then design their own maze. Robomind users create their maze maps in a text editor. However, to simplify the process for young pupils we use a standard spreadsheet. The pupils colour a 16 by 16 grid. The underlying formulae convert this into the text needed. Pupils can import this into Robomind to test their procedure to the limit! *Mark Dorling*

COMPUTER CONTROL OR IS THAT PROGRAMMING?

Mark Dorling recently presented two lectures to teacher trainees at Brunel University. The lectures focused on helping them understand how to identify the key concepts that underpin control, the learning journey that pupils travel and demystifying controls' curriculum position. Entitled 'Control or is that Computing', they were well received by all. The pioneering work done by the Digital Schoolhouse project at Langley Grammar School has been shared with ICT coordinators in 70 primary schools within the local authorities affiliated with Brunel's ITT partnership.

STEPPING UP TO THE CHALLENGE OF TEXT BASED PROGRAMMING

Small Basic is a highly accessible development environment that introduces beginners to programming in a fun an interesting way. It enables students to get results fast and that sense of achievement spurs them on to further development.



Being console based gives an immediacy and sense of control that my students love. They completely 'get' that the computer does exactly what they command, no more and no less. With a target audience of 10-16 year olds, my Year 7 students were able to put together a cheeky little questionnaire in less than half an hour, inadvertently learning about If Statements on the way.

Combining a friendly interface with simple language (consisting of only 14 keywords) and a rich set of libraries, Small Basic makes development an intriguing adventure. In a few lines of code, kids can create functioning programs involving standard concepts like conditions, loops, branching and sub-routines. The libraries give them access to clocks, sound, maths and image objects. Even stacks, arrays and file handling are addressed in a manageable way. However, to maintain simplicity it avoids concepts like type, object orientation and scope.

On the supporting Microsoft Small Basic website there are more than 20 lesson presentations (see example) and lots more help. Many lessons are game based and use contexts familiar to students such as Tetris. These lend themselves to club sessions for those who can't find time within the curriculum. You'll also find demos and code examples like the Sorting Algorithm Visualizer created by Zeven to graphically represent different sorting algorithms: a neat way of comparing their efficiencies, of use to all those teaching A Level Computing. There's plenty of room to extend Small Basic since its compiler is designed to allow the addition of external libraries developed in any .Net based language and compiled to a .Net assembly.

There is a thriving community. For example the fun '25 Line Program Challenge' produced a range of programs from Space Invaders to Sudoku Solvers, to Rock-Paper-Scissors. Whatever your interest, jump in and have a go – there's plenty of support available. Who knows what you may inspire in your kids. *Lyndsay Hope*

PROGRAMMING SESSIONS FOR TRAINEE TEACHERS

A programming summer school is being held at Anglia Ruskin University (in Chelmsford, Essex) this July for just-completing and just-starting PGCE ICT trainees. The course will run for 5 full days from July 25th to 29th. The intention is to give future ICT teachers with no experience of computer programming some basic programming skills and the confidence to develop these further. The course will use the Python language which has a growing educational following as more resources become available. The summer school is one of several initiatives by ITT institutions, funded by grants made available by the TDA. ***Sue Sentance***

GOOGLE'S LESSONS IN COMPUTATIONAL THOUGHT

Computational thinking involves a set of problem-solving skills and techniques that software engineers use to write programs. However, computational thinking is applicable to nearly any subject. Specific techniques include: problem decomposition, pattern recognition, pattern generalization to define abstractions or models, algorithm design, and data analysis and visualization. An important part of bringing computational thinking into the classroom is finding opportunities to highlight patterns; generalize rules from those patterns; and formalize the rules into simple programs. Helping students see the value and importance of this process by making it explicit is key.

Students who learn to think in computational terms begin to see a relationship between the different subjects they study. When a teacher can connect the data skills used in humanities or maths with the same data skills used in science, it reinforces their importance, and helps students understand that it's the same set of skills applied in different domains. The same is true with patterns and algorithms - it's the same thought process with different applications. It is a key skill for the 21st century, along with "reading, writing and arithmetic".

Using spreadsheets and Python programs, Google have collated over seventy lessons, oriented on maths and science topics which promote the approach. Aimed at secondary school level, from KS3 up, they provide a comprehensive set of stimuli, along with the associated programs, exercises and examples. Take a look at the introductory Python lesson to get a flavour of these excellent resources. Links in the supplement.

CAS WORKSHOPS FOR TEACHING 'A' LEVEL

Last term workshops were held in London and Manchester designed by CAS in association with Vital to help teachers of A-level Computing master some of the new topics on the AQA A2 Computing specification. Teachers became students for the day and tackled areas of the syllabus such as the complexity of algorithms (big-O, how fast can we compute something?), models of computation and computability (what cannot be computed?), graphs (how do we represent things?) and more.

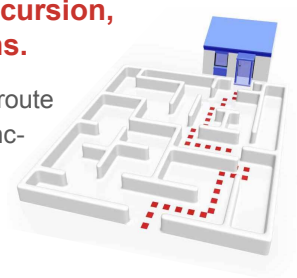
With the aid of a large number of presentations and supporting documents, Adrian Jackson and Stephen Hunt led attendees through the more complex components of the specification. For example, given any program and its input data, can we detect whether or not that program halts or not: the Halting problem. Adrian showed us how to prove that there is at least one program that cannot be written. The proof involving supposing the program existed, then feeding a small variation of it back into itself and showing this led to a paradox!

The sessions were very well received by all. I've since used the concepts and methods introduced to tie together a number of separate concepts, e.g. using a directed graph to represent programs, where a vertex represents a line of the program, and an edge between vertex A and vertex B represents the fact that the one line can be executed directly after another. This allows students to take simple programs of their own and create graphs to describe them, leading to discussions on complexity measures of programs. The chance to discuss such issues in greater depth was a unique and stimulating opportunity.

John Stout

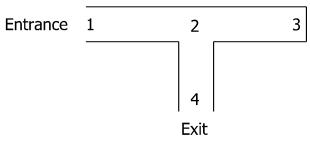
ADVANCED LEVEL EXPLORATIONS IN ROUTE FINDING WITH GRAPHS

Route finding offers plenty of scope for exploring many Computing concepts such as abstraction, data modelling, algorithm efficiency, record structures, arrays, lists, recursion, stacks, objects, procedure calling and graphs.

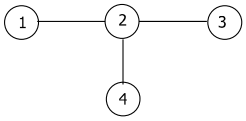


The picture shows a typical route finding task, finding a route through a maze from entrance to exit. How could the junctions, entrance, exit and pathways between these be represented if the task was delegated to a computer program?

First let's abstract away unnecessary detail and record the maze in a representation called a graph using the following rules for vertices and edges. A vertex can represent a starting / finishing point (entrance or exit), any dead ends and all the points in the maze where more than one path can be taken (junctions). Edges connect the vertices according to the paths in the maze.



To keep the description as simple as possible consider the "maze" pictured here. It has one entrance labelled 1, one dead end labelled 3, one exit labelled 4 and one junction labelled 2.



The diagram right shows the graph that models this is. The circles represent vertices or nodes and the interconnecting lines, edges. How can this graph be represented for route finding software? We can use an adjacency list as shown in the table. Vertex 1 is connected to vertex 2. Vertex 2 is connected to vertices 1, 3 and 4, etc.

Vertex	Adjacent vertices
1	2
2	1, 3, 4
3	2
4	2

This table can be represented by a one-dimensional array of records with each record storing the corresponding adjacent vertices (the first record 2, the second record 1, 3, 4 and so on). The structures for this are shown below.

```
ListType = Array[1..MaxNoOfAdjacentVertices] Of Integer;

VertexType = Record
               List : ListType;
               NoOfListVertices : Integer;
               State : String;
             End;

GraphType = Array[1..MaxNoOfVertices] Of VertexType;
```

Top right next page shows how a Graph array of the type GraphType would be initialized for the graph. The state field is needed so that when route finding the state of a vertex can be changed from "undiscovered" to "discovered".

A route from vertex 1 to vertex 4 can be found using the recursively defined algorithm also outlined on the next page, coded as procedure FindRouteToGoal with signature:

```
FindRouteToGoal(Var Graph : GraphType; CurrentVertex : Integer;
GoalVertex : Integer; Var Stack : StackType);
```

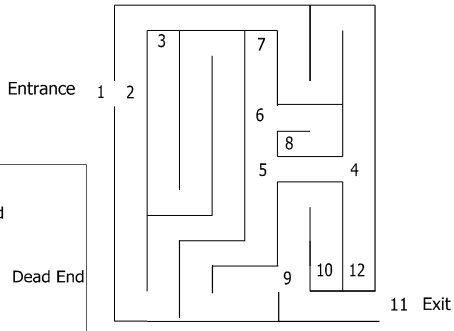
The Var refers to the parameter passing mechanism known as "call by address". Procedure FindRouteToGoal is called initially with CurrentVertex = 1 and GoalVertex = 4.

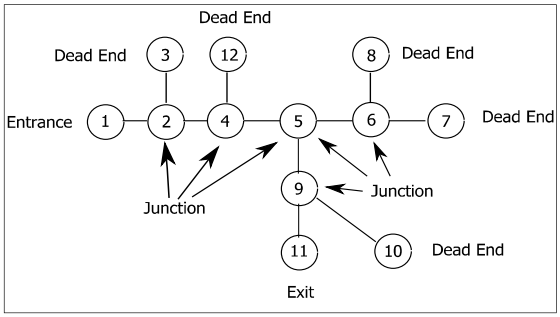
```
Graph[1].List[1] ← 2; Graph[1].NoOfListVertices ← 1; Graph[1].State ← 'undiscovered';

Graph[2].List[1] ← 1; Graph[2].List[2] ← 3; Graph[2].List[3] ← 4; Graph[2].NoOfListVertices ← 3; Graph
[2].State ← 'undiscovered';

Graph[3].List[1] ← 2; Graph[3].NoOfListVertices ← 1; Graph[3].State ← 'undiscovered';

Graph[4].List[1] ← 2; Graph[4].NoOfListVertices ← 1; Graph[4].State ← 'undiscovered';
```

```
Graph[CurrentVertex].State ← 'discovered';
Stack.Push (CurrentVertex);
If CurrentVertex = GoalVertex
  Then
    While Not Stack.Empty
      Do
        {
          Node ← Stack.Pop;
          Writeln(Vertex);
        }
  Else
    If Graph[CurrentVertex].NoOfListVertices <> 0
      Then
        For i ← 1 To Graph[CurrentVertex].NoOfListVertices
          Do
            If (Graph[Graph[CurrentVertex].List[i]].State = 'undiscovered')
              Then
                {
                  Vertex ← Graph[CurrentNode].List[i];
                  FindRouteToGoal(Graph, Vertex, GoalVertex, Stack);
                  Stack.Pop;
                }
```

Each time this procedure is called the number of the current vertex is pushed onto the stack. If a trial path fails to reach the goal vertex the stack is popped until a path not yet explored can be reached. If the goal vertex is reached, the stack will contain the vertices for a route between the starting vertex and the goal vertex but in the reverse order. This route is printed out before the execution of the procedure halts. If the goal cannot be reached the procedure will halt with the stack empty.

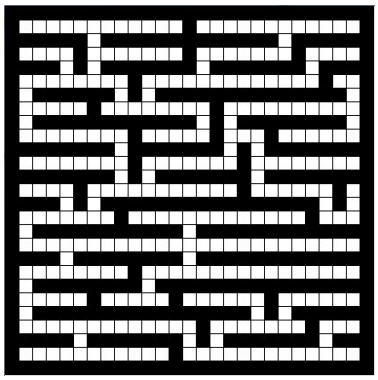
Here is a more complex maze. The equivalent graph is shown below.





Sketching the state of the stack as the algorithm above is hand traced is a useful exercise for students to try.

The work here could form the basis of several projects. For example, one such project could generate mazes of specified dimensions as shown in the example opposite. This could then be route searched using the algorithm given above. One application seen in project work is the design of climbing walls. There are plenty others. Route finding problems provide practical exemplification of many of the concepts studied in A Level Computing.



This article is based on the content of an Educational Computing Services Ltd course on teaching AQA A Level Computing and preparing candidates for COMP4 project work. See the web supplement for more details.

SIXTH FORM STUDENTS AT CAS CONFERENCES

The 2011 series of eleven CAS 6th form conferences taking place at venues throughout the UK, offered even more inspirational speakers than last year. Delivering engaging activities and promoting the excitement of Computer science, to over one thousand 16-18 year olds and over fifty accompanying teachers. Speakers from industry, business and academic life gave the school students insights into how varied a career in computing could be.

Neil Brown demonstrates using Kinect with Greenfoot at the Cockermouth conference

Feedback has been very positive with over 80% of teachers who responded stating that they were interested in developing computing activities as a supplement to the ICT curriculum. Over 90% said they would take students to a CAS conference next year. From the students, the following comment was indicative of the feedback received; “Good day, enjoyed it very much given the varied range of professions the speakers were in.” More than 50% stated that they were more likely to study computing at university having attended a CAS Conference. “I think overall it was a very interesting and informative day, and I gained a lot from it.”, said one student. *Claire Davenport*

STUDENTS INSPIRED BY APPS FOR GOOD INITIATIVE

Apps for Good is an award-winning programme where young people learn to create imaginative mobile apps that change their world. The course, which aims to fuel young peoples' interest in solving social and community problems with technology is already causing a stir.

The Central Foundation Girls School in Tower Hamlets is the first school in the UK to adopt the Dell Youth Connect supported Apps for Good course from charity CDI Europe, which recently won a Learning Without Frontiers award and is currently shortlisted for the Guardian Megas. Between April and June 2010, the course piloted with 9 young people aged 16-25 at High Trees Development Trust in South London. Now, in addition to running open courses for unemployed 16 to 25 year olds, the tech-education charity is set to roll out the programme in schools.

During the Apps for Good course, students go through a kind of entrepreneurial process whereby they identify what is wrong with their world before designing a way of fixing it with a mobile app. The course covers a broad range of areas, giving young people a foundation in entrepreneurship, community involvement, problem-solving and team work, as well as design and some technical skills. At Central Foundation Girls School, students aged 13-17 came up with five new ideas, including an app to improve communication between teachers and the large community of Bengali-speaking parents in the area.

Plans to open source Apps for Good online with funding from the Nominet Trust are currently underway. If you are interested in offering the Apps for Good course at your school, please get in touch with Iris Lapinski. Contact details and further links in the web supplement. *Iris Lapinski*

CODING A BETTER COUNTRY WITH YOUNG REWIRED STATE

Young Rewired State is a network of young developers set up as the philanthropic arm of Rewired State. If you teach keen, inventive coders make sure they know about YRS and encourage them to get involved.

Every year there is a week long event where YRS applicants and alumni gather in local businesses that have developers working in them. From Monday to Friday the young coders build applications, widget or websites (based on data held on data.gov.uk) with the guiding hand of the developers working in the businesses, each other and Rewired State mentors. On Friday afternoon everyone travels to London for a show and tell to press, government and industry. Read all about what we did last year on our website. YRS is growing fast, so spread the word.

This year we are aiming to find every person aged 18 or under in the UK who can code. We will find the nearest centre to you, so for now all we want are names and locations, plus how good you are. Don't worry, the point is to learn from some of the country's top developers and to show how awesome our burgeoning young geek talent.

There will be prizes, pizza and press. Afterwards, you never know! Last year, one of our YRSers ended up delivering her app to the Prime Minister and a cross government competition; many more have been employed or taken on internships; some have used the experience to set up their own businesses. After the week, students will be eligible to attend the Rewired State hack days, which can mean you could get to go to SXSW, Glastonbury or even Stephen Fry's Big Digital Day - and earn some money in the meantime. A hackday is best described as an intense piece of R & D built over one or two days. We take some of the UK's most talented developers (we recommend no fewer than 10 and have run events with more than 100), put them in a big room and they thrash out digital solutions to anything you choose to throw in their direction. At the end of the hack, this powerhouse of talent present prototypes demonstrating a range of solutions. Links to examples can be found in the supplement. If you'd like to kick start a new project please get in touch. As Helen Kilpatrick, Transparency Champion at the Home Office said; "Working with Rewired State stimulated us to think about new ways in which government data can be used – and inspired us to put even more data 'out there'." If you teach keen young coders, make sure they know what's happening, sign up and get excited. Young Rewired State 2011 will take place from the 1st to 5th August— it is an opportunity not to be missed. See the link in the supplement. *Emma Mulqueeny*

JUDGING GETS UNDERWAY FOR ANIMATION 11

Spare a thought for the judging team at Animation11. The deadline for entries to the well established schools animation competition arrived on 1 April. This year saw 886 entries from 158 schools, involving 1,180 students. The Awards Festival and Inspirational Computer Science Day will take place at the University of Manchester on 1st July. It will include hands on workshops and an inspirational talk; 'Computer Science Comes Alive!' by Computer Graphics expert Fred Gill of Electronic Arts.

TECHNICAL COMPUTING DAYS PROVE A BIG HIT WITH PUPILS

Higher education establishments can help enrich the experiences of school students. Rob Williams from UWE Bristol reports on how their well established Technology Workshops for schools can bring benefits to all participants.

When I returned to academic life in 1985 I decided to introduce a radical new degree, now called "BSc Computer Systems Integration". For many years this hybrid award, part hardware, part software, thrived and grew. However, our ability to recruit school pupils has, unfortunately steadily declined over recent years.

To attract UCAS applications and alert school pupils to what we now refer to as a career in "Technical Computing" we introduced the "Computer Systems Technology Schools Workshop" events. These run for two days, twice a year and attract around 100 students each day. For this years' events, the attendees chose two activities from a list of eight alternatives:

- Buggy programming in C,
- Games programming in Java,
- PC build and boot (Linux),
- Data recovery from hard disk,
- Configuring a home network,
- Soldering a circuit,
- Simulation using Flash,
- Digital electronic design.

The various workshop activities are based on some of our own first year lab practicals that have been adapted to suit students from Years 11 to 13. The workshop days are all structured to offer two 90 minute practical sessions, one in the morning and the second in the afternoon.

One worthwhile aspect of attending these workshop days is the opportunity to meet with our own students who have volunteered to assist with the practical lab work. A particularly successful arrangement is when sessions are chosen to fulfil the demands of various Key Stage indicators, which may otherwise be difficult to cover using the limited resources in school.

CST Schools Workshops are held in January and June. We have a regular contact list of 35 schools and colleges within the south-west region. Several of these originated with our own alumni who heroically chose to take up teaching upon graduation! Visiting groups vary in size from 70 to occasional senior pupils permitted to attend unaccompanied by teachers.

Support from local employers remains strong. We are proud to continue with the "Vocational Mission" which the previous Polytechnic institutions pioneered for UK higher education. Employers tell us there remains a desperate need for graduates with Computer Systems skills beyond programming desktop PCs. With the emergence of "intelligent" mobile phones, touch tablets and Cloud Computing, they now need education to introduce students to a range of tools and techniques to prepare for an unpredictable, exciting future.

Rob Williams

WORKSHOPS ORGANISED BY UNIVERSITY OF SURREY

On 12 April, the Faculty of Engineering and Physical Science at the University of Surrey held an introductory Java workshop for teachers. Participants were able to work at their own pace through practical examples with support from University staff. They are also planning two free web mashup workshops for school students on 30 June (Year 12) and 13 July (Year 11). The workshops offer hands-on experience of creating applications using Google Maps, and an appreciation of creating readily shareable photo albums. More details in the supplement.

TEENTECH INNOVATORS AND OTHER BCS EVENTS

On March 18, BCS Qualifications team joined up with BCS Berkshire branch to offer a green screen technology experience to over three hundred local 14 to 15 year old school students. The annual Teentech event offers an insight into careers in technology. The event was hosted by the former BBC "Tomorrow's World" presenter, Maggie Philbin, who said: "There is a dangerous complacency about the UK skills shortage in the industries of the future."

Also in March, BCS Bristol Branch organised 4 workshops on the background to mobile apps, and how to create, develop, and market them. Over 70 delegates attended. The workshop was kindly hosted by the City of Bristol College. Delegate Tony Clark said: "I was very impressed by the aspects covered by the speakers. We were given insights into the "big picture" covering the design of location-aware mobile applications through to the technical challenges of such applications."

During National Volunteer Week the BBC Coventry and Warwickshire Open Centre event took place on 23 Feb. Over 8's were invited to go along and take part in a workshop offering to help people with their software and mobile applications. Pictured above is Warwick University volunteer, Sam Edwards, giving help to Andrew Walker in one workshop.

CAS RESPONDS TO GOVT CALLS FOR CONSULTATION

Many of you will know about the National Curriculum Review, and many of you have contributed to conversations about it through the CAS discussion forum. John Woollard led a small group in developing a CAS/BCS response to the Call for Evidence, and ran a small survey of your views. At the recent face-to-face CAS working group meeting we discussed the draft extensively, and agreed the main points. A link to the final result is in the supplement. The submission differs only in that we split it up a little to match the form of the Review's questionnaire.

I hope you like it. In any large group — and CAS is, happily, now a large group with over 500 members, and the explicit support of BCS — there is bound to be a diversity of views, so one submission is never going to precisely reflect any one person's opinion. But we share a common sense of the thrill of computing, and a strong desire to share that excitement with our young people, and that is the number one message of the submission. If we have expressed it differently than you would have done, please forgive us!

The same link in the web supplement also gives our submissions to the Commons Select Committee (March) and the Royal Society enquiry (November). You will also find "Computing: a curriculum for schools" featured on page 2. It's been a busy few months, but a few months that have offered unprecedented opportunities for us to make our case at national level. A huge thank-you to all those who contributed, most especially John Woollard and Bill Mitchell. *Simon Peyton-Jones*

STEPPING UP A GEAR: ROUND UP OF RECENT CAS ACTIVITIES

The Computing At School group continues to grow, both in membership and influence. With it come increasing opportunities to develop initiatives to support teachers in getting computing back into the classroom.

It's been a busy time over the last term. Alongside the response to consultations featured (left), via the BCS Academy, CAS contributed significantly to the schools section of the Livingstone-Hope report. In July 2010 Ed Vaizey MP, Minister for Culture, Communications and the Creative Industries asked Ian Livingstone and Alex Hope, working with NESTA and Skillset to produce an independent report into the skills needed for school leavers and graduates to fully engage with the UK's world-class video games and visual effects industries. We were very pleased with the overall content of this report as it chimes with the message for more explicit Computer Science teaching in our schools. You can find links to several reports with similar messages in the supplement.

Getting the message across to policy makers is but one area of our work. More importantly, CAS seeks to support those on the ground who are making things happen in the here and now. Local hubs, where colleagues can meet to discuss and share ideas are springing up everywhere. Hertfordshire, West London and South Wales held there first meetings recently, whilst Swindon and Brighton have their inaugural meetings this term. Regional hubs are the life and soul of CAS. Those who have attended can testify to their benefits. If you would like to launch a hub in your area, Claire Davenport would love to help in getting it off the ground.

Professional development for teachers has become another key priority. Our first courses and workshops have used a variety of methods with the assistance of Vital. We have the opportunity to deliver courses to different audiences using various delivery styles. We need to know what you want and what works best.

Teachers have told us they want resources, training, pupil conferences and support. We want to provide as much as we can. Next month, our third annual teacher conference takes place. Please make every effort to come along and help shape the future. Educate, engage and encourage is the CAS mission, but as we've always said; there is no 'them', only us. Hope to see you at conference! *Simon Humphreys*

CAS TEACHER CONFERENCE 2011

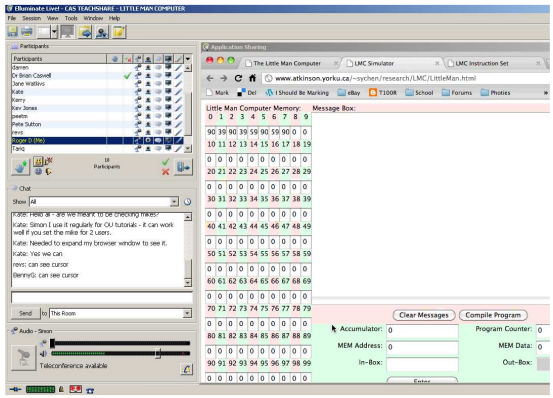
The final preparations are being made for the third annual CAS Teacher Conference. A "not to be missed" event for those looking for ideas, resources and inspiration. This year will be bigger than ever and includes a keynote from Karen Brennan on Scratch and other activities from the MIT Media Lab. We guarantee that you will return buzzing with new ideas for the academic year ahead. See back page for details.

PLANS IN PLACE FOR CAS SUMMER SCHOOL

The first CAS summer school is taking shape. In association with STEM, a 3-day school for approximately 20 teachers is currently being finalised. If you are keen to be at the cutting edge of curriculum transformation make sure you don't miss out. An intensive workshop packed with 'unplugged' activities, training in a variety of software tools and other resources to promote the teaching of computing at secondary schools. Sessions on computing theory and discussions about building the computing curriculum. Scheduled from 3rd to 5th August at the wonderful STEM Learning Centre in York. More details in the supplement.

SHARING IDEAS AND PRACTICE THROUGH VITAL TEACHSHARES

Teachshares allow participants to gather together online, view demonstrations of software and discuss possibilities in an informal environment. They're proving a popular mechanism for sharing good practice as Mark Clarkson explains.



I saw, via Twitter, that John Stout was presenting a Teachshare on Scratch BYOB (Build Your Own Blocks). I'd heard of BYOB but didn't really know much more. By clicking the link I was taken to Elluminate, a virtual meeting room provided by Vital, where the presenter shares their screen and describes what they're demonstrating. There's a chat window to ask questions, discuss the topic and post links and URLs. The meeting lasted an hour. People could drop in and out without the need to create an account and I learned an awful lot. Obviously you can't teach a group of newcomers every aspect of the software but I got a really good demonstration of what BYOB can do and why I should take a closer look.

I found it so worthwhile, that when Simon Humphreys asked for further volunteers, I put myself forward. One month later, I found myself in the kitchen; wife and young children upstairs "because Daddy is talking to people on the Internet", specifically about the Little Man Computer (see supplement). When I logged on there were a dozen people waiting for me, which felt really nice. I double checked the settings, shared my browser window (see above) and spent the next hour talking about the LMC, GCSE Computing, interpreters, machine code, assembly language and more.

It was really enjoyable – demonstrating a tool I'm familiar with and introducing it to new people. The feedback has been really positive. A CAS Teachshare is something I would heartily recommend that people attend, and have a go at presenting. You might think what you do or know is 'ordinary' or something that everyone else already knows, but you'll be surprised. We all spend a lot of time in our own little bubbles. Getting the opportunity to share ideas and tools with fellow enthusiasts in a supportive environment is a great way to develop your CPD. And in times when getting released for courses is proving more difficult, it's worth noting it's free!

MORE DEVELOPMENTS PLANNED BY VITAL

As Vital's range of online resources expands, we're developing a series of portals to help you go directly to what interests you. Each subject portal contains links to free resources and materials in the Vital Library, as well as course suggestions and subject-specific 'Top Ten Tips'. Contribute to the discussions and help Vital create a lively community of shared teaching experience.

If you're considering working towards a Masters degree, you will find Vital's courses can be used as a starting point. We are working with several universities to ensure that work completed through Vital can be developed in their programmes. Find out more in the supplement about using Vital as a stepping stone to Masters level credit, and if you are a university offering Masters level awards then please get in touch.

The educational landscape is changing. What's happening in your area? A recent event, Learn from the Kiwi Experience – explored how schools in New Zealand shared technical expertise through working in clusters. Local TeachMeets such as those held recently in Poole, Fulham, Birmingham or Winchester have proved very popular. Keep an eye on the Vital events calendar for more TeachMeets planned for the summer term. *Emma Blackburn*

INTRODUCTION TO JAVA PROGRAMMING COURSE

Using Greenfoot to Introduce Java, took place in March, including two face-to-face workshops. Between the sessions, using Elluminate software, Michael Kolling offered remote tuition. Richard Vickery commented; "I found the course very useful. It was intense but good to work to a time limit. It made me focus on tasks I would not have done if I was working on my own. Having face to face sessions at the start and end with six one hour on-line sessions in between worked well. I now feel more confident about teaching trainee teachers how to use Greenfoot in the classroom. Michael was very helpful and supportive." CAS hope to run another course soon. Keep your eye on the website for further details.

SWITCHEDON NEEDS YOU!

Are you organising an event, running a competition or after school club? Maybe you have a lesson idea or resource you want to share or review. SWITCHEDON aims to cover all the exciting initiatives taking place that promote computing in school. Make sure you let us know what you are doing, and help inspire others to do likewise.

A PAUSE FOR THOUGHT

In this classic river-crossing problem, three missionaries and three cannibals must cross a river in a boat which can carry two people, but on both banks missionaries cannot be outnumbered by cannibals (if they were, the cannibals would eat them.) The boat cannot cross the river without people on board and all missionaries and cannibals can row. How can you get all six across the river and how many trips will it take?

In its original form the problem is trivial. However the problem can be elaborated, for example one variation states that three missionaries with a single cannibal can convert him into a missionary; another that when you try to get four missionaries and cannibals across it becomes unsolvable.

This problem is a 'toy problem', of no intrinsic value but useful to illustrate a facet of a larger problem or explain a problem solving technique. Saul Amarel used it as an example of problem representation in artificial intelligence. It is also commonly used to demonstrate searching in AI including classic search algorithms such as breadth-first and depth-first. Answer, and a link to an online version in the supplement.

Lyndsay Hope

FACING UP TO FACES: CS4FN AT ROYAL SOCIETY SCIENCE LIVE

The team from cs4fn are preparing a fascinating exhibit at the Royal Society Summer Science Exhibition, coming in July 2011. The RSSSE is free and a fantastic day out in London, meeting working scientists and talking with them about their research.

Facing Up To Faces will be all about why faces matter. They are windows to our emotions and identities. Explore how our brain understands faces, experience the science and help our scientists create the next generation of life-changing software and robots. You'll be able to take part in interactive experiments, and we'll have some giveaways for students and teachers who come to our stall. You can find more information about planning a visit on the Royal Society's exhibition website.

Look out too for the next issue of Computer Science For Fun magazine in the early summer. It will be full of stories of shady deeds and mysteries, from crime to cryptography, all with a connection to computer science. We deal with interdisciplinary science too, highlighting ways in which computing links with other subjects. To make sure you receive the next issue of cs4fn, sign up for a free copy (or class set) by following the link to order a copy from our website. Links in the supplement. *Jonathan Black*

COMPUTING: THE NEXT GENERATION

Computing At School Third Annual Teacher Conference
Birmingham University: Friday 24th June (reception 23rd June)

The government are turning their attention to the National Curriculum; industry is calling for more Computing Science in the school curriculum; the Royal Society are conducting a report into the Computing and ICT in schools. Three excellent reasons for those who are passionate about computing as a subject to meet together and share ideas, resources and best practice. A day packed with meetings, discussions and activities guaranteed to inspire, educate and bring computing to life. Numbers are strictly limited so book your place as soon as possible. **http://casconf2011.eventbrite.com**

"I learnt more and connected with more interesting people than on any course that my school has previously forked out hundreds of pounds for"

COMPUTING AT SCHOOL
EDUCATE · ENGAGE · ENCOURAGE

CAS...JOIN CAS...JOIN CAS...JOIN CAS...JOIN CAS...JO

Computing at School was born out of our excitement with the discipline, combined with a serious concern that students are being turned off computing by a combination of factors. Our goal is to put the fun back into computing at school. Will you help us? Simply mail

membership@computingatschool.org.uk

Many thanks to the following for help with this issue of SWITCHEDON: *Miles Berry, Jonathan Black, Emma Blackburn, Kevin Bond, Mark Clarkson, Quintin Cutts, Claire Davenport, Roger Davies, Mark Dorling, Kathryn Harkup, Lyndsay Hope, Toby Howard, Simon Humphreys, Michael Kölling, Iris Lapinsky, Emma Mulqueeny, Simon Peyton-Jones, Sue Sentance, John Stout, Rob Williams.*

SWITCHEDON WEB SUPPLEMENT **www.computingatschool.org.uk**